WEEKLY REPORT – SINDHUJA CHADUVULA

This report documents the process of converting a traffic network from the Simulation of Urban Mobility(SUMO) XML format to OpenDrive, then to Lanelet2 and finally visualizing it using Common road and JSOM(Java OpenStreetMap Editor). The conversion facilitates the use of traffic network data across different simulation and design platforms, enhancing the capabilities for testing and developing autonomous vehicle algorithms.

Introduction

The advancement of autonomous vehicle technology necessitates the use of diverse simulation tools to model and analyze traffic networks. This report outlines the conversion of a SUMO-generated traffic network into formats compatible with autonomous driving simulation tools. SUMO's XML format, while robust for traffic simulations, requires conversion to OpenDRIVE for high-fidelity road network descriptions and to Lanelet2 for detailed lane-level information. CommonRoad Designer serves as the final platform for visualization and scenario editing.

- **SUMO**: An open-source, highly portable, microscopic and continuous road traffic simulation package designed to handle large road networks.
- **OpenDRIVE**: A standardized file format used to describe road networks in detail, including lane information, road signs, and traffic lights.
- **Lanelet2**: An advanced map format that provides a partitioned representation of road networks, focusing on the drivable areas and the rules that apply to them.
- **CommonRoad Designer**: A tool for creating, editing, and visualizing scenarios for autonomous driving simulations, compatible with various map formats, including Open Street Map (OSM).

Methodology

The conversion of traffic network data from SUMO to Lanelet2 via OpenDRIVE and CommonRoad involves a multi-step process that ensures the retention of critical road network information while transitioning between different formats. Each format serves a unique purpose in the simulation and analysis of traffic networks, necessitating careful conversion to maintain the integrity and usability of the data.

From SUMO to OpenDRIVE:

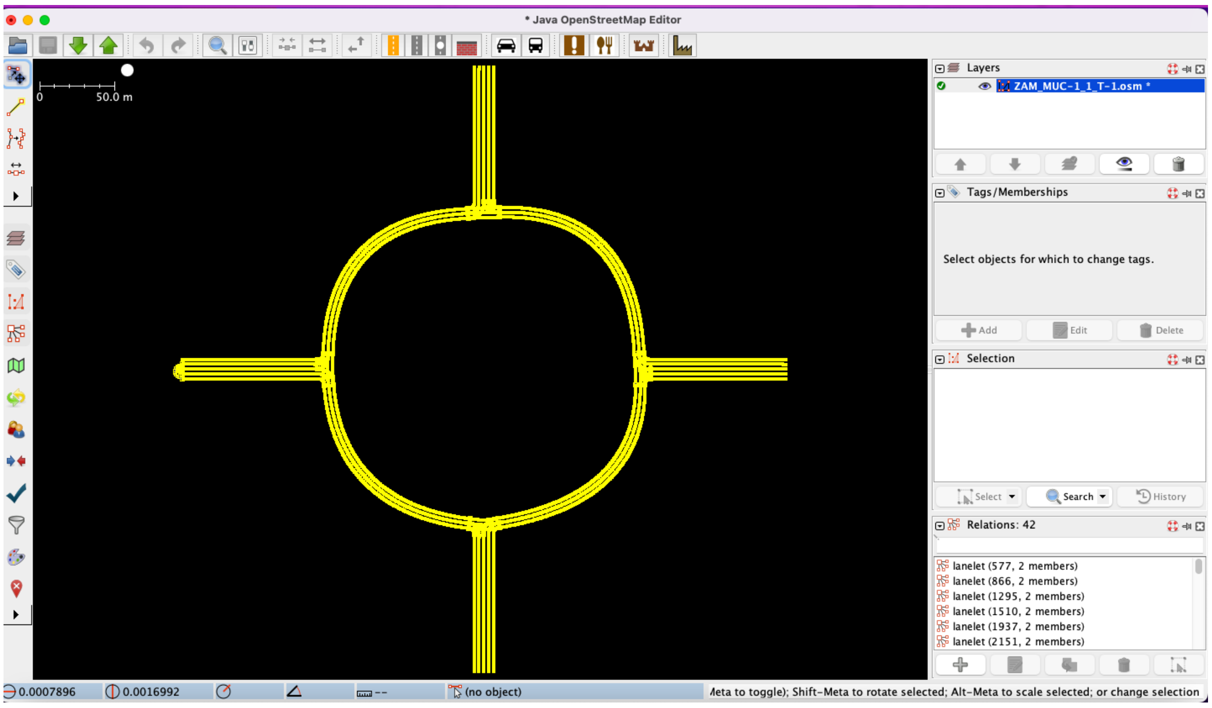
- Conversion Tool: Utilize the netconvert utility from SUMO to convert XML-based road network definitions into the OpenDRIVE format.
- Data Integrity: Ensure the SUMO network is properly prepared for conversion, adjusting parameters as necessary to maintain accuracy in the OpenDRIVE output.

From OpenDRIVE to CommonRoad:

- Import Process: Use CommonRoad's functionalities to read the OpenDRIVE file and transform it into a CommonRoad scenario, preserving road topology and traffic rules.
- Scenario Creation: Leverage CommonRoad's scenario creation capabilities to facilitate the visualization and editing of autonomous driving scenarios.

From CommonRoad to Lanelet2:

- Conversion Script: Apply a custom script or conversion tool to map CommonRoad scenario data to Lanelet2 primitives, including road and lane structures and traffic regulations.
- Format Readiness: Ensure the final Lanelet2 format is detailed and precise, suitable for use in advanced autonomous driving applications.


* Java OpenStreetMap Editor
0
50.0 m
Layers
ZAM_MUC-1_1_T-1.osm *
Tags/Memberships
Select objects for which to change tags.
Add
Edit
Delete
Selection
Select
Search
History
Relations: 42
lanelet (577, 2 members)
lanelet (866, 2 members)
lanelet (1295, 2 members)
lanelet (1510, 2 members)
lanelet (1937, 2 members)
lanelet (2151, 2 members)
0.0007896
0.0016992
(no object)
/eta to toggle); Shift-Meta to rotate selected; Alt-Meta to scale selected; or change selection

SCENARIO-1: Exploring potential predictions or classification models to determine vehicle types under specific configurations with a step length of 1sec

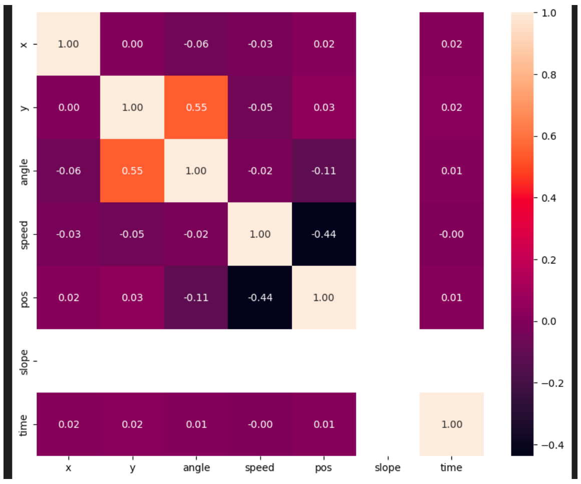
The features in the dataset are
id: This is a unique identifier for each vehicle instance in the simulation. It often contains both the type of vehicle and a unique number (e.g., flow_aggressive_E.0 is the ID for the first aggressive vehicle in the eastbound flow).
x and **y:** These are the spatial coordinates of the vehicle on the simulation grid or map, representing the vehicle's current position. The x coordinate typically represents the horizontal position, while the y coordinate represents the vertical position.
angle: This is the orientation of the vehicle in degrees, with 0 degrees usually representing east, 90 degrees north, and so on. It indicates the direction the vehicle is facing.
speed: This is the current speed of the vehicle in the simulation. The units are meters per second.
pos: This might represent the position of the vehicle along the lane it's currently in, it is measured in meters from a certain reference point, like the start of the lane or the last junction.
lane: This indicates the specific lane the vehicle is in, with the naming convention likely indicating direction and lane number.
slope: This could represent the gradient or incline of the road at the vehicle's current position, which can affect vehicle speed and behavior. However, in your dataset, this value

is 0.0 for all entries, which indicate a flat road or that slope isn't being considered in the simulation.
time: This is the simulation time at which the vehicle's state is recorded. It's likely measured in seconds from the start of the simulation. In your dataset, the initial time for all entries is 0.0, indicating the start of the simulation.

The types of vehicles and their count

Types	
Default	13603
Inexperienced	10918
distracted	10101
Speeder	9055
Aggressive	8951
Tailgater	7076
Passive	4227

The correlation between all the features is as shown below:



Model Descriptions

Random Forest: Random Forest is an ensemble learning method that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes of the individual trees. It is known for its robustness and accuracy, as it combines the predictions of several trees, which can individually capture different aspects of the data.
Gradient Boosting: Gradient Boosting is another ensemble technique that builds trees one at a time, where each new tree helps to correct errors made by previously trained trees. It uses a gradient descent algorithm to minimize the loss when adding new models.
Deep Neural Network (DNN): DNNs are complex neural networks with multiple hidden layers that can model intricate patterns in data. They are particularly good at capturing nonlinear relationships but require large amounts of data and computational power to train effectively.
Multilayer Perceptron (MLP): An MLP is a class of feedforward artificial neural network that consists of at least three layers of nodes: an input layer, a hidden layer, and an output layer. MLP utilizes a backpropagation algorithm for training the network and can capture complex relationships in the data.

K-Nearest Neighbors (KNN): KNN is a simple, instance-based learning algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions). KNN has been successful in a large number of classification and regression problems, including those that are nonlinear.

In our simulation, we applied various machine learning models to predict vehicle types based on simulation data with a step length of one second. The models included Random Forest, Gradient Boosting, Deep Neural Networks, MLP (Multilayer Perceptron), and KNN (K-Nearest Neighbors). Among these, the Random Forest model achieved the highest accuracy at 36%. While this figure is not particularly high in absolute terms, it is instructive to analyze why the Random Forest outperformed its counterparts in this context.

```
              precision    recall  f1-score   support

           1       0.36      0.41      0.39      4055
           2       0.39      0.36      0.37      2714
           3       0.33      0.28      0.30      2745
           4       0.32      0.31      0.32      3090
           5       0.37      0.42      0.39      3212
           6       0.34      0.30      0.32      2106
           7       0.38      0.39      0.38      1258

    accuracy                           0.36     19180
   macro avg       0.36      0.35      0.35     19180
weighted avg       0.35      0.36      0.35     19180

Accuracy: 0.35604796663190824
```

Despite its relative superiority, the 36% accuracy indicates that there is significant room for improvement. This could involve more sophisticated data preprocessing, feature selection, and engineering, as well as hyperparameter optimization. Additionally, the low accuracy across all models suggests that the problem at hand is complex and may benefit from more advanced modeling techniques or additional data that can help to distinguish between vehicle types more effectively.
In conclusion, while the Random Forest model did not achieve high absolute accuracy, its performance relative to other models suggests it is better suited to this dataset under the current conditions.

SCENARIO 2 - Exploring potential predictions or classification models to determine vehicle types under specific configurations with a step length of 0.1sec

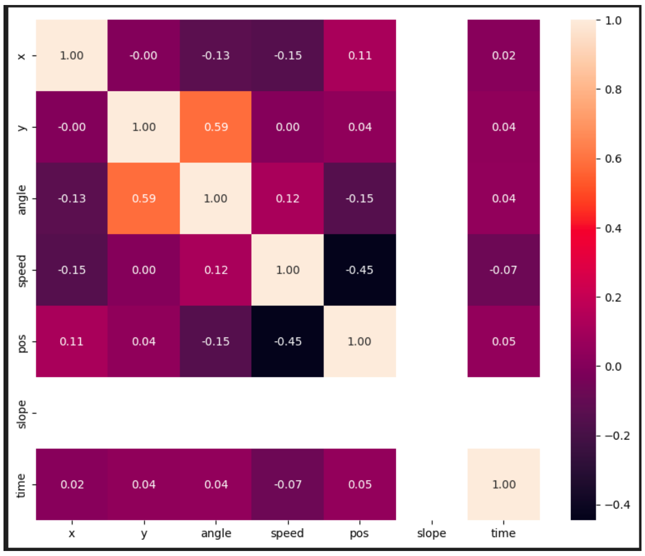
The features in the dataset are
id: This is a unique identifier for each vehicle instance in the simulation. It often contains both the type of vehicle and a unique number (e.g., flow_aggressive_E.0 is the ID for the first aggressive vehicle in the eastbound flow).
x and **y:** These are the spatial coordinates of the vehicle on the simulation grid or map, representing the vehicle's current position. The x coordinate typically represents the horizontal position, while the y coordinate represents the vertical position.
angle: This is the orientation of the vehicle in degrees, with 0 degrees usually representing east, 90 degrees north, and so on. It indicates the direction the vehicle is facing.
speed: This is the current speed of the vehicle in the simulation. The units are meters per second.

pos: This might represent the position of the vehicle along the lane it's currently in, it is measured in meters from a certain reference point, like the start of the lane or the last junction.
lane: This indicates the specific lane the vehicle is in, with the naming convention likely indicating direction and lane number.
slope: This could represent the gradient or incline of the road at the vehicle's current position, which can affect vehicle speed and behavior. However, in your dataset, this value is 0.0 for all entries, which indicate a flat road or that slope isn't being considered in the simulation.
time: This is the simulation time at which the vehicle's state is recorded. It's likely measured in seconds from the start of the simulation. In your dataset, the initial time for all entries is 0.0, indicating the start of the simulation.

The types of vehicles and their count

Types	
Default	130498
Inexperienced	90974
distracted	79088
Speeder	63844
Aggressive	54895
Tailgater	50724
Passive	28061

The correlation between the features



Model Descriptions

In our comprehensive analysis, we evaluated several machine learning models to classify vehicle types based on simulation data. The models tested included Random Forest, Gradient Boosting, MLP (Multilayer Perceptron), KNN (K-Nearest Neighbors), and SVM (Support Vector Machine). The Random Forest classifier emerged as the top performer, achieving an accuracy of 84%.

Random Forest Classifier:
Ensemble Approach: Random Forest leverages the power of ensemble learning, where multiple decision trees vote on the final classification. This approach inherently provides a balance between bias and variance, leading to more accurate and stable predictions.
Feature Handling: It is capable of handling a large number of input features and maintaining accuracy even when most of the features are not informative, which can be an advantage in complex datasets.
Non-Linearity: The model can capture complex, non-linear relationships without the need for transformation, which is crucial in datasets with intricate interactions between variables.

```
              precision    recall  f1-score   support

           1       0.97      0.97      0.97     38818
           2       0.76      0.75      0.76     19163
           3       0.90      0.83      0.86     16517
           4       0.81      0.85      0.83     27274
           5       0.77      0.81      0.79     23974
           6       0.76      0.74      0.75     15374
           7       0.75      0.72      0.73      8306

    accuracy                           0.84    149426
   macro avg       0.82      0.81      0.81    149426
weighted avg       0.84      0.84      0.84    149426

Accuracy: 0.8399475325579217
```

Gradient Boosting:
Gradient Boosting is another ensemble technique that builds trees sequentially, with each tree trying to correct the errors of the previous one. While powerful, it may not have performed as well due to overfitting or not being tuned adequately to handle the specific noise and structure of the data.
MLP (Multilayer Perceptron):
MLPs are deep learning models that are highly flexible and capable of modeling complex relationships. However, they require extensive hyperparameter tuning and large amounts of data to generalize well, which might explain their lower performance.

The Random Forest classifier's robust performance, with an accuracy of 83%, underscores its suitability for complex traffic simulation data. Its ability to handle large datasets with numerous features, coupled with its resistance to overfitting, makes it an excellent choice for this application. Future research may focus on further optimization of this model, exploring more complex ensemble methods, or even deep learning architectures as the dataset grows in size and complexity.